Digital Circuits Theory - Laboratory

Academic year	Laboratory exercises on	Mode of studies	Field of studies	Supervisor	Group	Section
2020/2021	Wednesday 15:30 – 17:00	SSI	Informatics	KP	1	3

Report from Exercise No 6

Performed on: 28.10.2020

Exercise Topic: Selected arithmetic circuits

Performed by:

Dawid Grobert
Julia Boczkowska

Purpose of the exercises

The aim of the exercise was to learn about such elements as: adder, comparator. More generally, we were dealing with arithmetic combinations and sequential systems performing arithmetic operations.

1 Description of the first task

Task 1

Using prepared by yourself test data (pairs of input values and the expected outcomes for the operation), verify the working mode of 4-bit adder with respect to the used representation of binary numbers, whether it adds numbers:

a) with sign, in Sign-Magnitude notation,

b) with sign, in Diminished Radix Complement notation,

c) with sign, in Radix Complement notation,

d) without sign.

1.1 Analysis of the task

My test data will be:

$$\begin{aligned} (X)_{10} = 2 \longrightarrow (X)_2 = &\ 010 \\ (Y)_{10} = 4 \longrightarrow (Y)_2 = &\ 100 \end{aligned}$$

1.2 Sign Magnitude calculations

a) with sign in Sign-Magnitude notation

For X + Y

$$z_s = x_s = y_s \longrightarrow z_s = 0$$

$$\begin{array}{c} \left.\begin{array}{c} |X| = 010 \\ |Y| = 100 \end{array}\right\}+ \\ \hline |Z| = 110 \end{array}$$

$$(Z)_2^{SM} = 0.110$$

Expected result

For (-X) + (-Y)

$$z_s = x_s = y_s \longrightarrow z_s = 1$$

$$\begin{array}{c} \left.\begin{array}{c} |X| = 010 \\ |Y| = 100 \end{array}\right\}+ \\ \hline |Z| = 110 \end{array}$$

$$(Z)_2^{SM} = 1.110$$

Expected result

For X + (-Y)

$$x_s \neq y_s$$

$$\begin{array}{c} \left.\begin{array}{c} |X| = 010 \\ |\overline{Y}| = 011 \end{array}\right\}+ \\ \hline |\overline{Z}| = 101 \end{array}$$

no carry

$$(Z)_2^{SM} = 1.010$$

Expected result

$$z_s = y_s = 1$$

For (-X) + Y

$$x_s \neq y_s$$

$$\begin{array}{c} \left.\begin{array}{c} |X| = 010 \\ |\overline{Y}| = 011 \end{array}\right\}+ \\ \hline |\overline{Z}| = 101 \end{array}$$

no carry

$$(Z)_2^{SM} = 0.010$$

Expected result

$$z_s = y_s = 0$$

1.2.1 The circuit diagram of Sign-Magnitude Notation adder

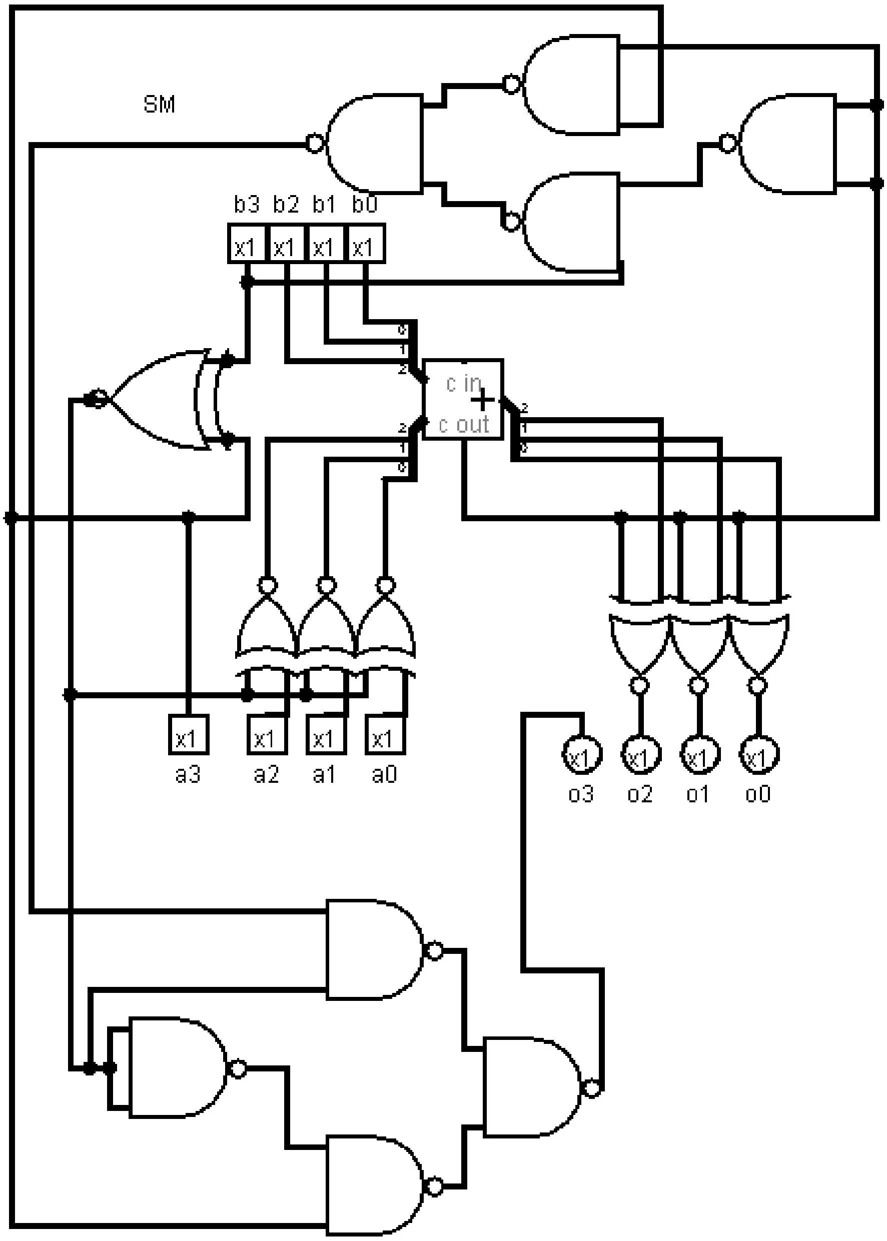


Figure 1: 4-bit Adder for Sign-Magnitude Notation

1.3 Diminished Radix Complement Calculations

b) with sign, in Diminished Radix Complement notation

$$\begin{array}{l}\left.\begin{array}{r}(X)_2^{DRC} = 0.010\\ (Y)_2^{DRC} = 0.100\end{array}\right\}+\\ \hline (Z)_2^{DRC} = 0.110\end{array}$$

Expected result

$$\begin{array}{l}\left.\begin{array}{r}(-X)_2^{DRC} = 1.101\\ (-Y)_2^{DRC} = 1.011\end{array}\right\}+\\ \hline \left.\begin{array}{r}1 \longleftarrow\ 1.000\\ \hookrightarrow\ \ 1\end{array}\right\}+\\ \hline (Z)_2^{DRC} = 1.001\end{array}$$

Expected result

$$\begin{array}{l}\left.\begin{array}{r}(-X)_2^{DRC} = 1.101\\ (\ \ Y)_2^{DRC} = 0.100\end{array}\right\}+\\ \hline \left.\begin{array}{r}1 \longleftarrow\ 0.001\\ \hookrightarrow\ \ 1\end{array}\right\}+\\ \hline (Z)_2^{DRC} = 0.010\end{array}$$

Expected result

$$\begin{array}{l}\left.\begin{array}{r}(\ \ X)_2^{DRC} = 0.010\\ (-Y)_2^{DRC} = 1.011\end{array}\right\}+\\ \hline (Z)_2^{DRC} = 1.010\end{array}$$

Expected result

1.3.1 The circuit diagram of Diminished Radix Complement Notation adder

The 4-bit adder for Diminished Radix Complement notation is as follows:

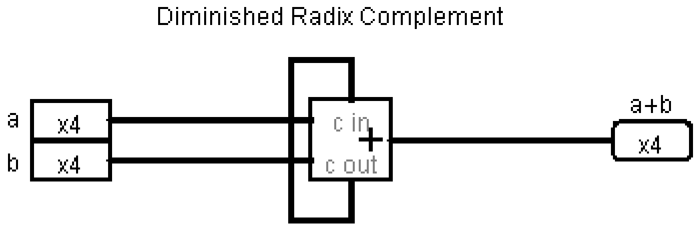


Figure 2: 4-bit Adder for Diminished Radix Complement notation

1.4 Radix Complement Calculations

c) with sign, in Radix Complement notation

$$\begin{array}{l}\left.\begin{array}{r}(X)_2^{RC} = 0.010\\ (Y)_2^{RC} = 0.100\end{array}\right\}+\\ \hline (Z)_2^{RC} = 0.110\end{array}$$

Expected result

$$\begin{array}{l}\left.\begin{array}{r}(-X)_2^{RC} = 1.110\\ (-Y)_2^{RC} = 1.100\end{array}\right\}+\\ \hline \cancel{1}\longleftarrow\ (Z)_2^{RC} = 1.010\end{array}$$

Expected result

$$\begin{array}{l}\left.\begin{array}{r}(-X)_2^{RC} = 1.110\\ (\ \ Y)_2^{RC} = 0.100\end{array}\right\}+\\ \hline \cancel{1}\longleftarrow\ (Z)_2^{RC} = 0.010\end{array}$$

Expected result

$$\begin{array}{l}\left.\begin{array}{r}(\ \ X)_2^{RC} = 0.010\\ (-Y)_2^{RC} = 1.100\end{array}\right\}+\\ \hline (Z)_2^{RC} = 1.110\end{array}$$

Expected result

1.4.1 The circuit diagram of Radix Complement Notation adder

The 4-bit adder for Radix Complement notation is as follows:

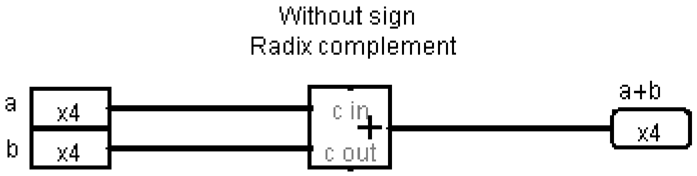


Figure 3: 4-bit Adder for Radix Complement notation

1.5 Without Sign calculations

d) without sign

$$\begin{array}{l} \left.\begin{array}{l} |X| = 010 \\ |Y| = 100 \end{array}\right\}+ \\ \hline |Z| = 110 \end{array}$$

Expected result

1.5.1 The circuit diagram of numbers without sign adder

The 4-bit adder for numbers without sign is as follows:

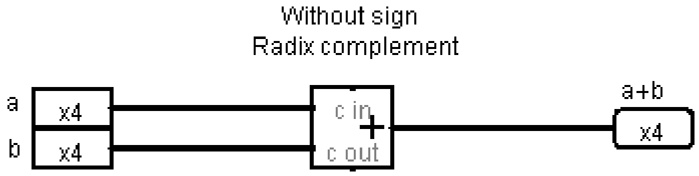


Figure 4: 4-bit Adder for unsigned numeric type

This space has been intentionally left blank.

2 Description of the second task

Task 2

Using prepared by yourself test data (pairs of input values and the expected outcomes of the operation), verify the working mode of 4-bit comparator with respect to the used representation of binary numbers, whether it adds numbers:

a) with sign, in Sign-Magnitude notation,

b) with sign, in Diminished Radix Complement notation,

c) with sign, in Radix Complement notation,

d) without sign.

2.1 Analysis of the task

My test data will be:

$$\begin{array}{rcl} (X)_{10} = 2 & \longrightarrow & (X)_2 = \ 010 \\ (Y)_{10} = 4 & \longrightarrow & (Y)_2 = \ 100 \end{array}$$

a) with sign in Sign-Magnitude notation

Expected result:

$$\left.\begin{array}{r} (X)_2^{SM} = 0.010 \\ (Y)_2^{SM} = 0.100 \end{array}\right\} \quad X < Y$$

$$\left.\begin{array}{r} (-X)_2^{SM} = 1.010 \\ (-Y)_2^{SM} = 1.100 \end{array}\right\} \quad X > Y$$

$$\left.\begin{array}{r} (-X)_2^{SM} = 1.010 \\ (Y)_2^{SM} = 0.100 \end{array}\right\} \quad X < Y$$

$$\left.\begin{array}{r} (X)_2^{SM} = 0.010 \\ (-Y)_2^{SM} = 1.100 \end{array}\right\} \quad X > Y$$

b) with sign, in Diminished Radix Complement notation

Expected result:

$$\left.\begin{array}{r} (X)_2^{DRC} = 0.010 \\ (Y)_2^{DRC} = 0.100 \end{array}\right\} \quad X < Y$$

$$\left.\begin{array}{r} (-X)_2^{DRC} = 1.101 \\ (-Y)_2^{DRC} = 1.011 \end{array}\right\} \quad X > Y$$

$$\left.\begin{array}{r} (-X)_2^{DRC} = 1.101 \\ (Y)_2^{DRC} = 0.100 \end{array}\right\} \quad X < Y$$

$$\left.\begin{array}{r} (X)_2^{DRC} = 0.010 \\ (-Y)_2^{DRC} = 1.011 \end{array}\right\} \quad X > Y$$

c) with sign, in Radix Complement notation

Expected result:

$$\left.\begin{aligned}(X)_2^{RC} &= 0.010\\(Y)_2^{RC} &= 0.100\end{aligned}\right\}\quad X < Y$$

$$\left.\begin{aligned}(-X)_2^{RC} &= 1.110\\(-Y)_2^{RC} &= 1.100\end{aligned}\right\}\quad X > Y$$

$$\left.\begin{aligned}(-X)_2^{RC} &= 1.110\\(Y)_2^{RC} &= 0.100\end{aligned}\right\}\quad X < Y$$

$$\left.\begin{aligned}(X)_2^{RC} &= 0.010\\(-Y)_2^{RC} &= 1.100\end{aligned}\right\}\quad X > Y$$

d) without sign

Expected result:

$$\left.\begin{aligned}(X)_2 &= 010\\(Y)_2 &= 100\end{aligned}\right\}\quad X < Y$$

2.2 Sign-Magnitude notation comparator circuit diagram

The 4-bit comparator for Sign-Magnitude notation is as follows:

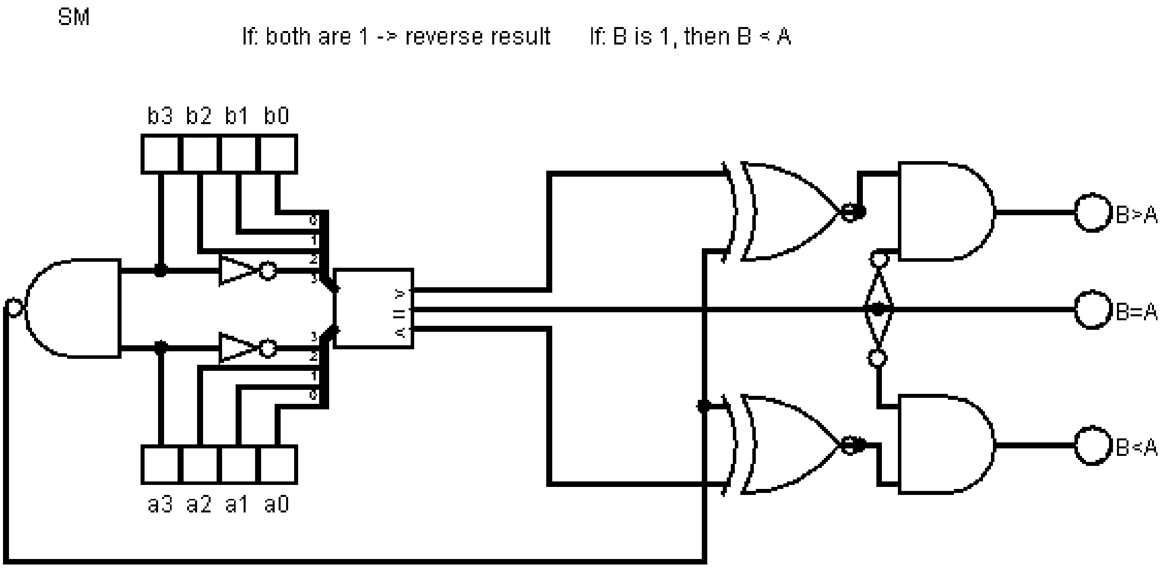


Figure 5: 4-bit comparator for Sign-Magnitude notation

2.3 Diminished Radix Complement comparator circuit diagram

The 4-bit comparator for Diminished Radix Complement notation is as follows:

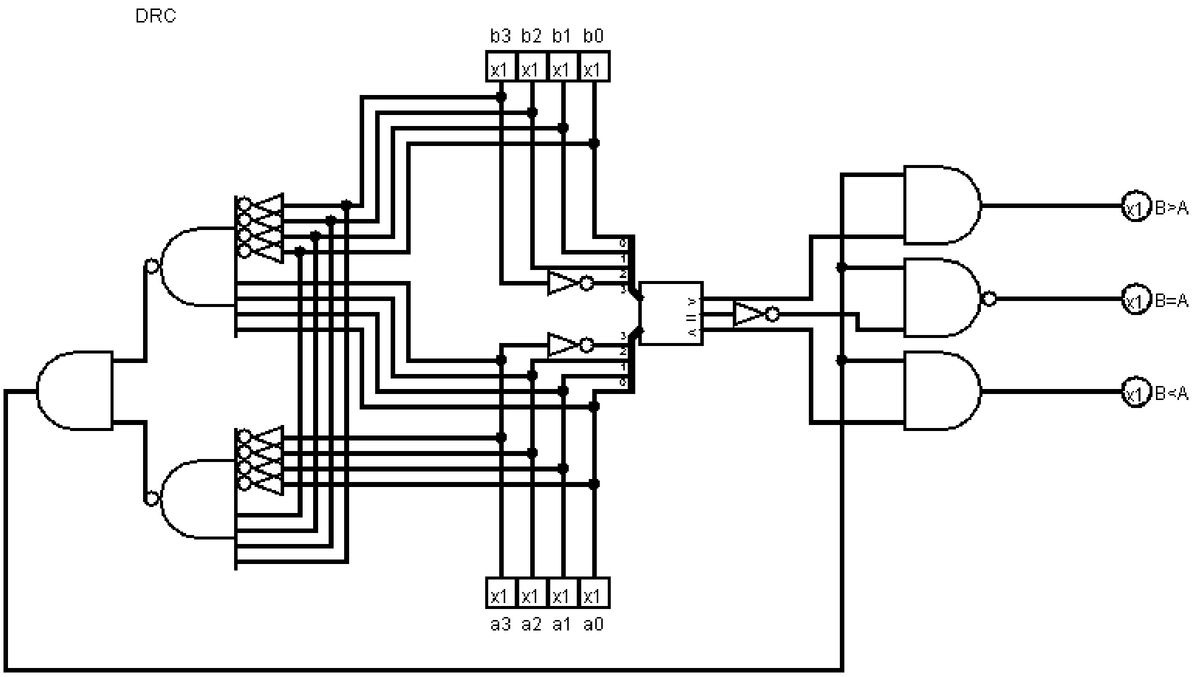


Figure 6: 4-bit comparator for Diminished Radix Complement notation

2.4 Radix Complement notation comparator circuit diagram

The 4-bit comparator for Radix Complement notation is as follows:

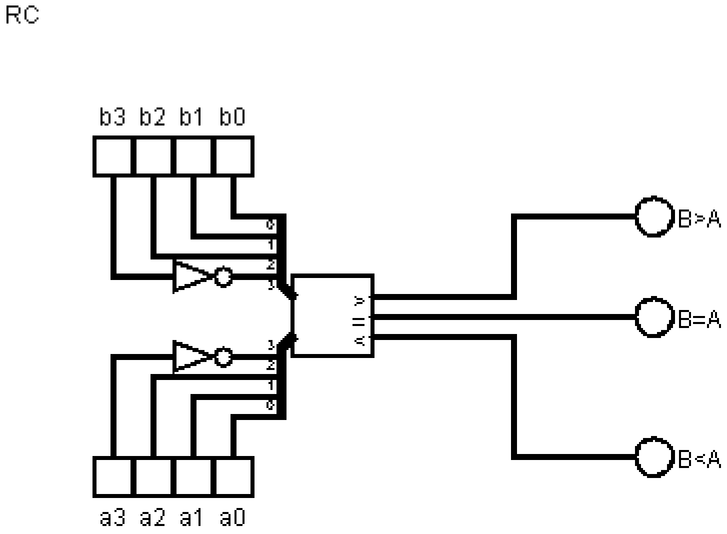


Figure 7: 4-bit comparator for Radix Complement notation

2.5 Circuit diagram for comparator of numbers without sign

The 4-bit comparator for numbers without sign is as follows:

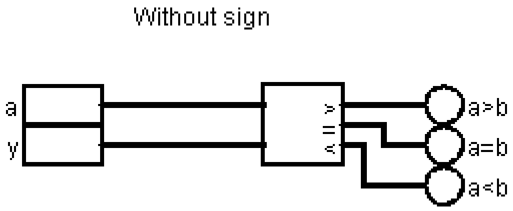


Figure 8: 4-bit comparator for unsigned numeric types

This space has been intentionally left blank.

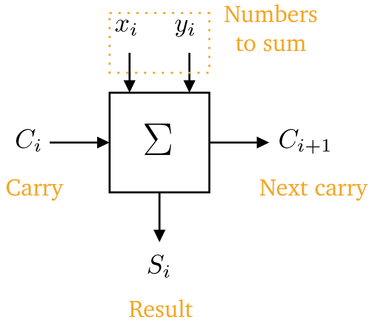
3 Description of the third task

Task 3

Obtain full 1-bit adder using NAND gates.

3.1 Analysis of the task

The circuit can be presented as follows:



3.2 Implementation

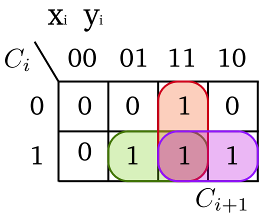
We can also very easily imagine and fill a similar Karnaugha grid. Truth Table can be helpful here.

C_i \ $x_i\ y_i$	00	01	11	10
0	00	01	10	01
1	01	10	11	10

$C_{i+1}\ S_i$

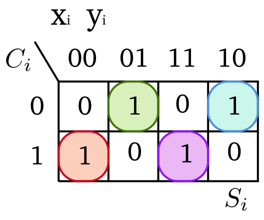
x_i	y_i	C_i	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

And read out individual groups from it:



C_i \ $x_i\ y_i$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

C_{i+1}



C_i \ $x_i\ y_i$	00	01	11	10
0	0	1	0	1
1	1	0	1	0

S_i

$$C_{i+1} = C_i\, y_i + x_i\, y_i \;+\; x_i\, C_i$$

$$S_i = C_i\, \overline{x_i}\, \overline{y_i} + \overline{C_i}\, \overline{x_i}\, y_i + C_i\, x_i\, y_i \;+\; \overline{C_i}\, x_i\, \overline{y_i}$$

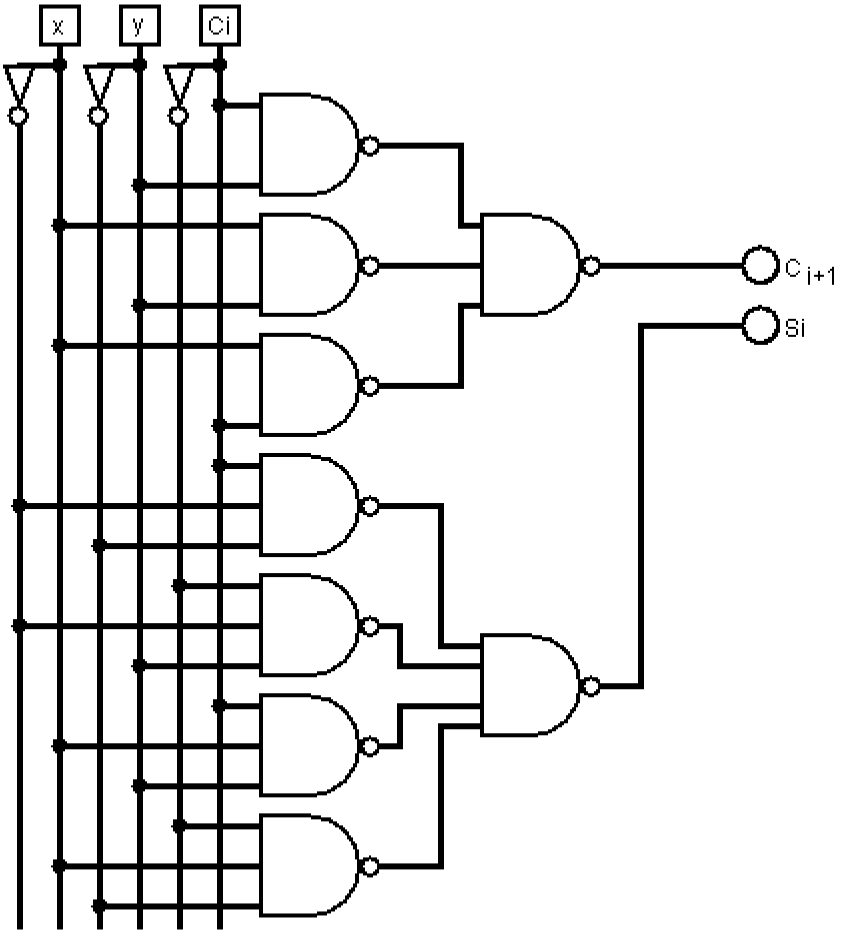
Implementation with NANDS:

$$C_{i+1} = \overline{\overline{C_i\ y_i} \times \overline{x_i\ y_i}\ \times\ \overline{x_i\ C_i}}$$

$$S_i = \overline{\overline{C_i\ \overline{x_i}\ \overline{y_i}} \times \overline{\overline{C_i}\ \overline{x_i}\ y_i} \times \overline{C_i\ x_i\ y_i}\ \times\ \overline{\overline{C_i}\ x_i\ \overline{y_i}}}$$

3.3 Circuit diagram

Finally, we get the given circuit:



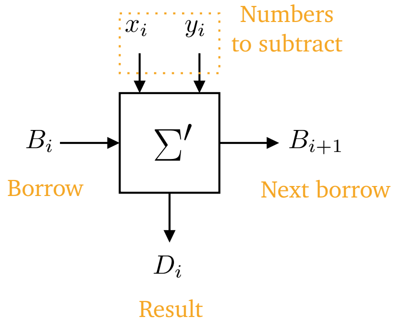
4 Description of the fourth task

Task 4

Obtain full 1-bit subtractor using NAND gates.

4.1 Analysis of the task

This type of task is even a mirror image of the previous task. The solution is almost the same.



4.2 Implementation

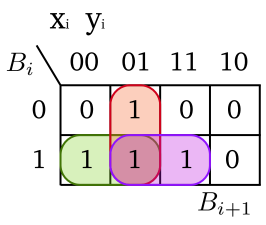
Again, the easiest way to complete the Karnaugh map is with the first filled-in truth table.

B_i \ $x_i\ y_i$	00	01	11	10
0	00	11	00	01
1	11	10	11	00

$B_{i+1}\ D_i$

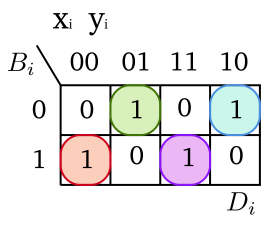
x_i	y_i	B_i	B_{i+1}	D_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Now we select the groups and we can see that D_i is identical to the summator's S_i, and B_{i+1} is literally a mirror image of the summator's map for C_{i+1}.



B_i \ $x_i\ y_i$	00	01	11	10
0	0	1	0	0
1	1	1	1	0

B_{i+1}



B_i \ $x_i\ y_i$	00	01	11	10
0	0	1	0	1
1	1	0	1	0

D_i

$$B_{i+1} = B_i\,\overline{x_i} + \overline{x_i}\,y_i + B_i\,y_i$$

$$D_i = B_i\,\overline{x_i}\,\overline{y_i} + \overline{B_i}\,\overline{x_i}\,y_i + B_i\,x_i\,y_i\ +\ \overline{B_i}\,x_i\,\overline{y_i}$$

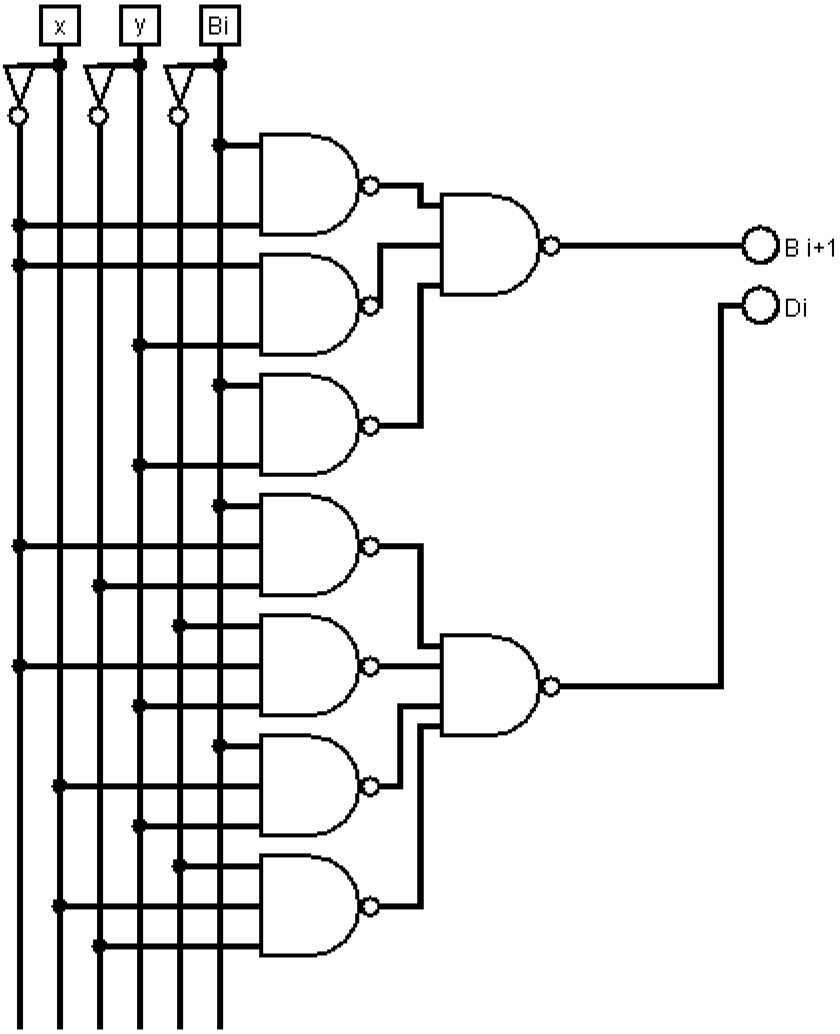
Implementation with NANDS:

$$B_{i+1} = \overline{\overline{B_i\ \overline{x_i}} \times \overline{\overline{x_i}\ y_i} \times \overline{B_i\ y_i}}$$

$$S_i = \overline{\overline{B_i\ \overline{x_i}\ \overline{y_i}} \times \overline{\overline{B_i}\ \overline{x_i}\ y_i} \times \overline{B_i\ x_i\ y_i}\ \times\ \overline{\overline{B_i}\ x_i\ \overline{y_i}}}$$

4.3 Circuit diagram

Finally, we get the given circuit:



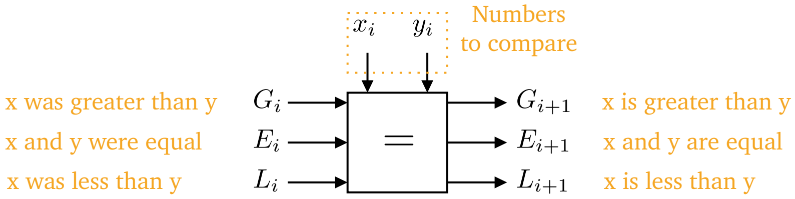
5 Description of the fifth task

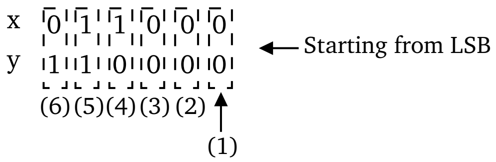
Task 5

Obtain full 1-bit comparator with comparison starting from the least significant bits.

5.1 Analysis of the task

Again, let's start the task with a short analysis and a certain graphic representation of the layout





(1) At this point the circuit thinks that the numbers are equal

(2) Again, 0 is equal to 0. Circuit still thinks the numbers are equal

(3) There is no other option for the circuit to know that the overall numbers are not equal. It still assumes that the numbers are equal.

(4) Finally some numbers differ. As x_4 is 1, and y_4 is 0 the circuit at this moment knows that x is bigger.

(5) Now we have two equal ones. However, from the previous comparison, the circuit still knows that x is somehow bigger.

(6) Now y_6 is 1 and x_6 is 0. This means that y is definitely bigger. This is the final comparison.

5.2 Implementation

This time without a truth table and with a short visual representation of the operation of this comparator. This is definitely enough to create a Karnaugh grid.

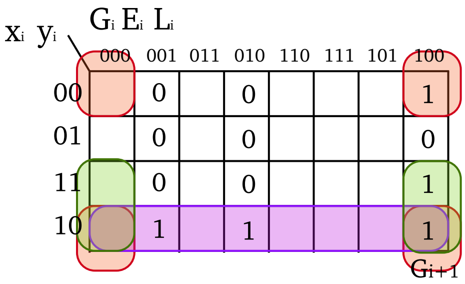
G_i E_i L_i

x_i y_i \	000	001	011	010	110	111	101	100
00		001		010				100
01		001		001				001
11		001		010				100
10		100		100				100

$G_{i+1}E_{i+1}L_{i+1}$

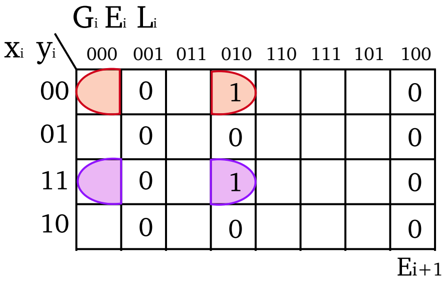
From this large Karnaugh map we get three Karnaugh maps.

I select the groups and derive the implementation using the AND, and OR gates.



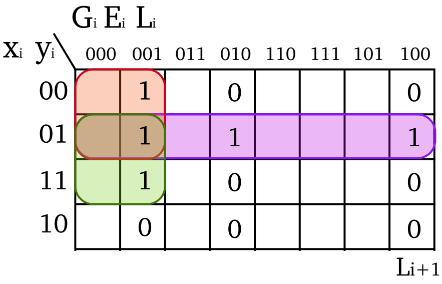
$x_i\ y_i \backslash G_i\ E_i\ L_i$	000	001	011	010	110	111	101	100
00		0		0				1
01		0		0				0
11		0		0				1
10		1		1				1

Gi+1



$x_i\ y_i \backslash G_i\ E_i\ L_i$	000	001	011	010	110	111	101	100
00		0		1				0
01		0		0				0
11		0		1				0
10		0		0				0

Ei+1



$x_i\ y_i \backslash G_i\ E_i\ L_i$	000	001	011	010	110	111	101	100
00		1		0				0
01		1		1				1
11		1		0				0
10		0		0				0

Li+1

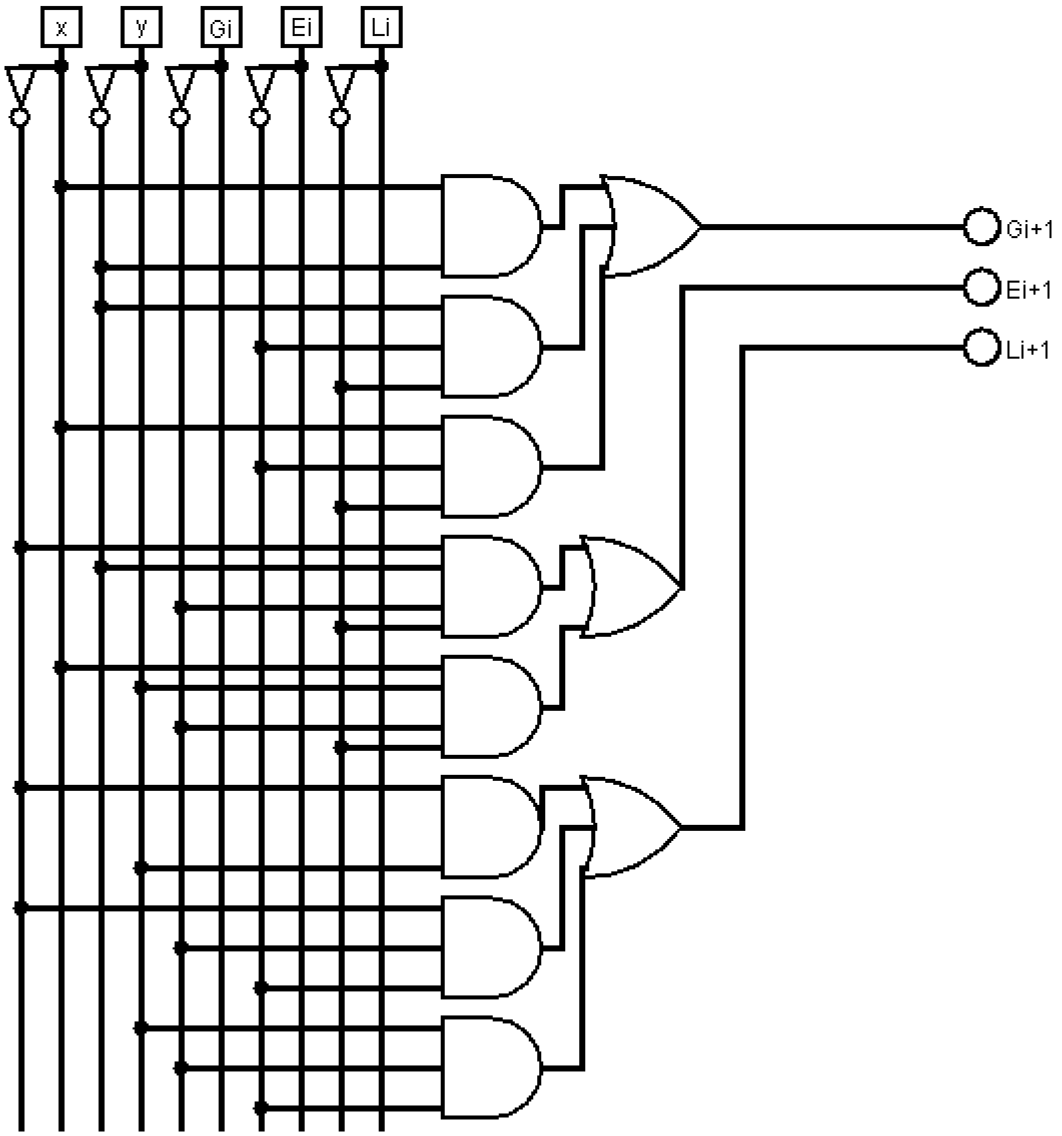
$$G_{i+1} = x_i\,\overline{y_i}\ +\ \overline{y_i}\,\overline{E_i}\,\overline{L_i}\ + \\ +\ x_i\,\overline{E_i}\,\overline{L_i}$$

$$E_{i+1} = \overline{x_i}\,\overline{y_i}\,\overline{G_i}\,\overline{L_i}\ + \\ +\ x_i\,y_i\,\overline{G_i}\,\overline{L_i}$$

$$L_{i+1} = \overline{x_i}\,y_i\ +\ \overline{x_i}\,\overline{G_i}\,\overline{E_i}\ + \\ +\ y_i\,\overline{G_i}\,\overline{E_i}$$

5.3 Circuit diagram

Finally, we get the given circuit:



6 Conclusions

The biggest surprise was definitely the first two tasks - comparators and adders. Definitely their implementation gave a lot of trouble, and left a lot of confusion. However, the rest of the tasks did not seem to be quite problematic - although I made a mistake many times in tasks 3, 4, 5 by connecting the Z_{i+1} outputs to Z_i inputs, which in this matter remain completely independent of each other.